Mobiilipelin toteuttaminen Unity-kehitysympäristössä

Android-käyttöjärjestelmälle

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulun opinnäytetyö

Mediatekniikan koulutusohjelma

Hämeen ammattikorkeakoulu, Riihimäki, kevät 2016

Mikko Erkkilä

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Riihimäki
Mediatekniikan koulutusohjelma

Tekijä Mikko Erkkilä **Vuosi** 2016

Työn nimi Mobiilipelin toteuttaminen Unity-kehitysympäristössä

TIIVISTELMÄ

Mobiilipelimarkkinoiden kasvaessa yleinen kiinnostus mobiilipelien kehitykseenkin on kasvussa. Tämän opinnäytetyön tarkoituksena on tutustua mobiilipelin toteuttamiseen käyttäen Unity-kehitysympäristöä, sen sisältämine työkaluineen. Projekti toteutettiin käyttämällä hyväksi Unityn laajaa käyttäjäkuntaa keskustelupalstoineen sekä internetistä löytyvien ohjeiden suurta kirjoa.

Tavoitteena on saada aikaan toimiva mobiilipelidemo, omaamatta alkeita syvempää ohjelmointitaitoa. Opinnäytetyössä käydään läpi, mitä vaaditaan Unityssä mobiilipelin toteuttamiseen ja julkaisuun, silmälläpitäen erityisesti Android-alustaa. Tutkitaan miten pelinkehitys onnistuu ilman aikaisempaa kokemusta ja todetaan kirjoittajan saaman opin määrä. Tutustutaan myös muutamaan vaihtoehtoiseen mobiilipelin kehitysalustaan ja mobiilipelien historiaan lyhyesti.

Työn tuloksena todetaan mobiilikehitykseen vaadittavan työn määrän mittavuus sekä ohjelmointitaidon kehittyminen projektin aikana.

Avainsanat Mobiilipelit, Android, peliohjelmointi,

Sivut 28 s.

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Riihimäki
Degree Programme in Media Technology

Author Mikko Erkkilä **Year** 2016

Subject of Bachelor's thesis Mobile Game Development in Unity

ABSTRACT

The mobile game market is on the rise and that causes more interest towards mobile game development. The goal of this thesis is to get to know how mobile game development can be done using Unity and its additional tools. The project will be executed with the help of the broad Unity Community and the vast amount of tutorials and other helpful information found online.

The goal is to create a functioning mobile game demo without having more than a basic knowledge of programming. The thesis will point out what exactly is needed in creating and publishing a mobile game using Unity, targeting especially Android. The thesis will reveal how much knowledge the writer gets while developing a game without previous knowledge of the subject. The thesis will also take a look at a few alternative game development environments and review the history of mobile gaming.

As a result of this research we will notice how much work mobile game development requires and how the programming knowledge of the writer improves.

Keywords Mobile games, Android, game development,

Pages 28 p.

SISÄLLYS

TERMIT JA LYHENTEET

Android	Android	Googlen omistama mobiilikäyttöjärjestelmä.
Asset	Resurssi	Unityn termi erilaisille tarvittaville resursseille. Esimerkiksi grafiikat, peliobjektit tai äänet yms.
Asset Store	Resurssikauppa	Unityn resursseja tarjoava kauppapaikka.
Google Play Store	Google Play Store	Googlen omistaman Android-käyttöjärjestelmän ohjelmien kauppapaikka.
SDK	SDK	Software Development Kit eli ohjelmistojen kehitys tarvikkeet yhdessä paketissa.
Scene	Kohtaus	Yhtä aikaa ladatuista peliobjekteista koostuva tila. Esimerkiksi yksi pelin taso.
Scene-näkymä	Kohtaus näkymä	Unityn käyttöliittymän osa, jossa näytetään peliobjektit joita kyseiseen sceneen on ladattuna.
Hierarchy	Hierarkia	Unityn käyttöliittymän osa, joka sisältää kaikki kyseiseen sceneen ladatut peliobjektit.
Inspector	Tarkastaja	Unityn käyttöliittymän osa, jossa peliobjektien määritykset ja asetukset näkyvät.
Script	Ohjelmakoodi	Ohjelmakoodi tai komentosarja, joka määrittelee jonkinlaista toimintaa.

Prefab	Esivalmistettu	Objekti jonka määritykset on tallennettu yhteen tiedostoon uudelleenkäyttöä varten.
Build	Koontiversio	Projektista pakattu jakelutiedosto.
Tutorial	Tutoriaali	Kurssi tai sarja ohjeita.
GameObject	Peliobjekti	Mikä tahansa pelissä toimiva osa.
MonoDevelop	MonoDevelop	Unityn mukana toimitettava ja suositeltava ohjelmointiympäristö.
Canvas Scaler	Piirtoalueen skaalaaja	Elementtien koon hallintaan käytettävä Unityn tarjoama ohjelmakoodi.
Unity Remote	Unity Remote	Unityn tarjoama mobiilikehitykseen tarkoitettu esikatselutyökalu.
APK-tiedosto	APK-tiedosto	Android Application Package, eli Android ohjelma tiedosto. Android sovellusten tiedostotyyppi

1 JOHDANTO

Mobiilipelimarkkinoiden kasvaessa kiinnostus mobiilipelien kehitykseenkin on kasvussa. Itseäni mobiilipelit ja niiden toteuttaminen ovat kiehtoneet jo pidemmän aikaa ja tästä syystä halusin niiden toteuttamiseen perehtyä hieman tarkemmin. Mobiilipeliyritykset myös työllistävät huomattavan määrän tekniikanalan opiskelijoita, mikä lisäsi mielenkiintoani aiheeseen tulevaisuutta ajatellen.

Opinnäytetyön tarkoituksena on tutustuttaa lukija mobiilipelin toteuttamiseen Unityllä ja tähän vaadittaviin toimenpiteisiin, keskittyen eritoten Android-alustalle kehittämiseen. Samalla toivon itse oppivani mobiilipelien kehityksestä sekä yleisesti pelien ohjelmoinnin rakenteesta. Oman heikon ohjelmointitaustani takia, on mielenkiintoista nähdä kuinka ohjelmointitaitoni kehittyvät tätä projektia toteuttaessani.

Alussa tutustutaan lyhyesti mobiilipelien historiaan sekä nykyiseen tilanteeseen. Tutustutaan myös muutamaan monista vaihtoehtoisista mobiilikehitysalustoista, kuten Unreal Engine ja Construct2. Tämän jälkeen käydään läpi Unityn perusominaisuuksia ja niiden käyttöä mobiilipeliä kehitettäessä.

Käytännönosuudessa toteutetaan toimiva pienimuotoinen mobiilipelidemo, ja tutustutaan mobiilipelien kehittämiseen vaadittaviin askeliin Unity-kehitysympäristössä. Käydään läpi tärkeimmät käytetyt toteutusmekaniikat ja muu kokonaisen pelin luomiseen tarvittavat lisäykset. Selvitetään myös lyhyesti mitä Unityllä kehitetyltä mobiilipeliltä vaaditaan, jotta se on julkaistavissa myös Googlen virallisessa kauppapaikassa, Play Storessa.

2 MOBIILIPELIT

Tässä luvussa käsitellään mobiilipelien historiaa ja nykyaikaa. Tutustutaan eri alustojen kehittymiseen ja nykypäivän uusiin tekniikkoihin.

2.1 Mobiilipelien historia

Ensimmäiset mobiilipelit löysivät tiensä puhelinlaitteisiin vuonna 1994. Pitkään luultiin, että ensimmäinen mobiilipeliä tukenut laite olisi Tanskassa valmistettu Hagenuk MT-2000, mutta sittemmin on väitetty, että IBM olisi ehtinyt ensin Simon-nimisellä älypuhelimellaan. MT-2000 laitteessa tuli mukana julkaisijan oma variaatio Tetris-pelistä kun taas IBM:n Simon-laitteella pystyi pelaaman Scramble-nimistä kuvanmuodostuspeliä. (Sager, 2012; Kuorikoski, 2015.)

Kolme vuotta myöhemmin Nokia julkaisi hitiksi nousseen versionsa meille suomalaisille matopelinä tutusta Snake (Blockade) pelistä, joka näkyy kuvassa 1. Nokia 6110 -laitteelle julkaistu matoseikkailu aloitti mobiilipelien uuden aikakauden ja Snake on uusine versioineen edelleen yksi tunnetuimmista mobiilipeleistä. (Unger & Novak, 2011.)

Kuva 1. Suomalaisille matopelinä tuttu Snake-mobiilipeli. (Osborn, 2015)

Muutamia vuosia myöhemmin mobiilipelimarkkinat uudisti WAP (Wireless Application Protocol), joka mahdollisti pelien lataamisen laitteelle, sekä verkon välityksellä useamman pelaajan samanaikaisen pelaamisen. Yksi ensimmäisistä mobiililaitteista joka tuki WAP-selaamista ja missä selain esiasennettuna toimitettiin, oli Nokia 7110. WAP mahdollisti pienimuotoisen internetselauksen, mutta laitteiden ominaisuuksien puitteissa tämä ei ollut kovinkaan kummoinen kokemus. (Wikipedia Nokia 7110, n.d.)

WAPin seuraajina tunnetaan J2ME (Java 2 Micro Edition) sekä BREW (Binary Runtime Enviroment for Wireless). J2ME:stä tuli suuri menestys Euroopassa, kun taas BREW oli enemmän amerikkalaisten ja aasialaisten kehittäjien mieleen. Nokia 3410 sekä Siemens M50 olivat ensimmäiset markkinoilla pärjänneet J2ME:tä tukeneet puhelinmallit. Samoihin aikoihin mobiilipelit saivat myös värit ensimmäisen massatuotetun värinäyttöpuhelimen, Sony Ericsson t681i:n mukana. (Phonearena, 2011.)

Nokia julkaisi vuonna 2003 tiedon tulevasta N-Gage puhelimestaan, jonka oli tarkoitus mullistaa jälleen kerran mobiilipelimarkkinat. Näin ei kuitenkaan käynyt, vaikkakin pelit laitteelle olivat kehittyneitä ja laitteella pelaaminen hieman käytännöllisempää kuin tavallisella, ei pelikäyttöön suunnitellulla puhelinlaitteella. N-Gage osoittautui Nokialle suureksi taloudelliseksi katastrofiksi. Nokia virheistään oppineena, alkoi keskittyä laitteidensa mukana tulleisiin Series40 ja Series60 -käyttöjärjestelmiin. (Phonearena, 2011.)

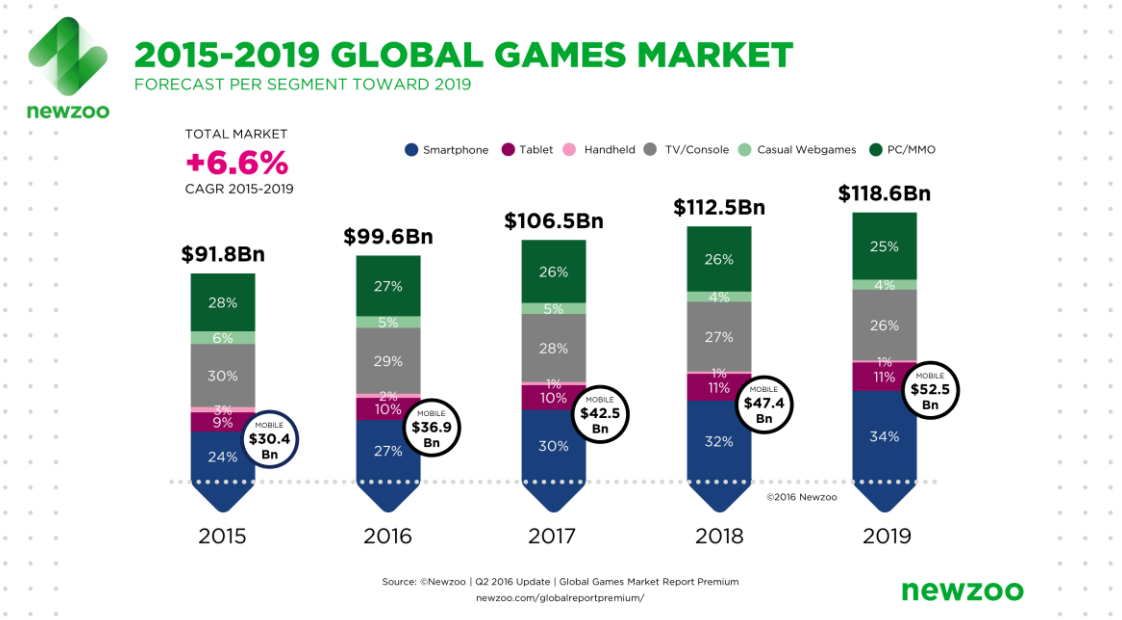
Mobiililaitteet kehittyivät hurjaa vauhtia ja kuluttajat alkoivat vaatia enemmän ja enemmän mobiilipeleiltä. Seuraavan suunnan mobiilipelimaailmalle näytti Apple, julkaistessaan ensimmäisen iPhonen joka mahdollisti pelien kontrolloinnin kosketusnäytöllä sekä laitteen kiihtyvyysanturia hyväksi käyttäen. Apple julkaisi samoihin aikoihin myös oman App Store -nimeä kantavan kauppapaikkansa, jonka kautta kuluttajien oli mahdollista ostaa

mobiilipelejä suoraan pelinkehittäjiltä, eikä teleoperaattorin tai muun palvelun tarvinnut enää olla välikätenä. (Wright, 2009; Wikipedia iPhone, n.d.)

2.2 Nykyaika

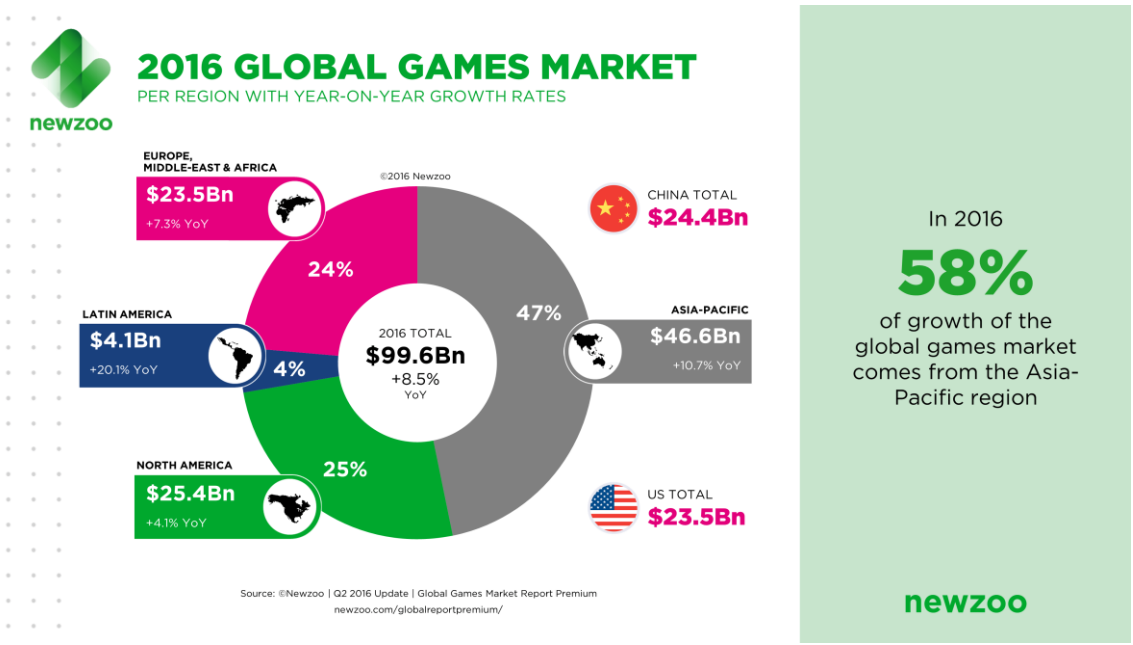
Nykypäivänä älypuhelinmarkkinat ovat massiiviset, ja niin on myös laitteille valmistettujen pelien määrä. Lähinnä Applen IOS, Googlen Android sekä Microsoftin Windows (Phone) käyvät kovaa taistoa älypuhelinkäyttöjärjestelmien herruudesta. Myös pelien kehittäjien kesken kilpailu on kovaa, koska nykypäivänä lähes kuka tahansa voi alkaa itse kotikoneellaan toteuttamaan mobiilipelejä, vaikka markkinoille asti. Monenlaiset kehitystyökalut mahdollistavat luovien ja tekniikkatietoisten ihmisten toteuttaa pelejä yllättävän helposti.

Mobiilipelaamisen suosio on jatkuvassa kasvussa ja rahallisesti se on osittain ohittanut PC- sekä konsolipelaamisen. Mobiilipelaamisen markkinoiden ennustetaan kasvavan edelleen ja jättävän muut alustat muutaman vuoden sisällä taakseen, kuten kuvasta 2 näkyy.



Kuva 2. Maailmanlaajuisten pelimarkkinoiden ennuste vuoteen 2019 asti. (Newzoo, 2016)

Maailmanlaajuisesti videopelimarkkinat ovat massiiviset. Aasia ja Tyynenmeren alue hallitsee noin puolta pelimarkkinoiden kokonaisliikevaihdosta. Kuvassa 3 näkyy kuinka eri maanosat ovat osallisina pelimarkkinoissa. Monen peliyrityksen suurin liikevaihto tuleekin Aasian ja Tyynenmeren alueelta, eikä niinkään Euroopasta tai Yhdysvalloista kuten saatetaan luulla.



Kuva 3. Maanosien osuus kansainvälisistä pelimarkkinoista. (Newzoo, 2016)

Mobiilipelimarkkinoille tuotetaan päivittäin uusia hittipelejä, joissa uudet innovaatiot ja teknologian kehittyminen näkyvät selkeästi. Yksi mobiilipelien isoimmista haasteista on pelien ohjaaminen eli kontrollit. Uusia tapoja kontrolloida, kehitetään ja innovoidaan jatkuvasti. Suomalaisen Fingersoftin Fast Like a Fox -pelissä esimerkiksi kontrolloidaan pelihahmoa naputtamalla puhelimen takaosaa, jolloin puhelimen kiihtyvyysanturi kertoo pelille, koska hahmo liikkuu ja kuinka nopeasti. Teknologian uusimmat tuulet, kuten virtuaalilasit ovat jo vahvasti näkyvissä mobiilipelimarkkinoilla. (Mobile Industry Review, 2016; Crawley, 2016; Fingersoft, n.d.)

Suomi tunnetaan maailmalla kovana mobiilipelien tuottajamaana, ja moni App Storen ja Googlen Play Storen Top 10 listalle yltävistä peleistä, onkin suomalaista tekoa. Rovion Angry Birds, Supecellin Clash of Clans (Boom Beach, Hay Day, Clash Royale yms.) sekä oululaisen Fingersoftin Hill Climb Racing ovatkin olleet kaikki mobiilipelilistojen kärkisijoilla.

3 TEKNINEN VERTAILU

Tässä luvussa tutustutaan muutamaan vaihtoehtoiseen kehitysalustaan ja pelimoottoriin. Tehdään myös valinta kehitys- sekä mobiilialustasta.

3.1 Kehitysalustan ja pelimoottorin valinta

Pelinkehitystä aloittaessa on kehitysalustan valinta syytä tehdä huolella. Kehitysalustoja on tarjolla satoja erilaisia, eikä ole olemassa yhtä joka olisi ylitse muiden. Vaikka jokin alusta olisi sopinut toiseen projektiin täydellisesti, voi se silti olla erilaista projektia toteuttaessa täysin hyödytön.

3.1.1 Unity

Unity on yksi tämän hetken suosituimmista pelinkehitysalustoista ja pelimoottoreista. Syynä tähän on sen laaja ilmainen perusversio, jonka kuka tahansa voi ottaa käyttöön täysin maksutta. Unity 5:n julkaisu toi mukanaan Personal Editionin joka sisältää lähes kaiken sen, minkä maksullinen Pro-versiokin. Unityn ilmaisversiolla on myös mahdollista julkaista projektejaan ilman lisenssimaksuja, kunhan tilin omistaja ei tienaa yli 100 000 $ vuodessa. (Unity Eula, n.d.)

Unityn ilmaisversio takaa sen, että uudet kehittäjät uskaltavat ainakin kokeilla alustaa ja sen myötä päätyvät sitä monesti, ainakin jossain määrin käyttämään. Unity toimii monelle pelistudiolle ainakin testausalustana, jolla uusia ideoita pääsee nopeasti kokeilemaan, kuluttamatta niiden toteuttamiseen liikaa resursseja. Unityllä on myös oma kauppapaikka (Asset Store), josta kehittäjät voivat hankkia lisäosia Unityyn tarpeen mukaan, esimerkiksi juuri testausta varten. (Unity Asset Store, n.d.)

Unity käyttää ohjelmointikielinään UnityScriptiä, C#- sekä Boo-kieliä. Näistä suosituin on C# jonka suurin osa Unityn käyttäjäkunnasta on ottanut ohjelmointikielekseen. UnityScript joka muistuttaa JavaScriptiä, on toiseksi suosituin käytetyistä ohjelmointikielistä ja Boo vähiten käytetty. Unityn koodi referenssi (Script Reference) viittaa näistä ainoastaan kahteen suositumpaan, mikä aiheuttaa sen että Boo-kielen käyttö on erittäin vähäistä.(Unity Script Reference, n.d.)

Unityn koodi referenssin lisäksi Unity tarjoaa referenssin kanssa yhdessä toimivan manuaalin sekä Learn-osion sivustollaan, mistä löytyy huomattava määrä ohjeita erilaisten pienten malliprojektien toteuttamiseen. Näiden lisäksi Unityn suuren käyttäjämäärän takia, internet on pullollaan erilaisia ohjeita erilaisiin projekteihin ja omissa projekteissa esiintyviin ongelmiin. Unityn käyttäjien foorumi (Community) tarjoaa myös paljon apua kehittäjille. (Unity Learn, n.d.; Unity Community, n.d.)

Unityn suosio perustuu myös sen laajaan alustatukeen. Tällä hetkellä Unity tukee yli kahtakymmentä eri alustaa aina pöytätietokoneesta älytelevisioon. Helppokäyttöisyys ja hyvä tukijärjestelmä ovatkin Unityn parhaita myyntivaltteja.

3.1.2 Construct 2

Construct 2 on Scirra-nimisen yrityksen tuottama HTML5-pohjainen pelieditori. Construct 2 mahdollistaa nopean pelin luomisen käyttämällä editorin omaa, raahaa ja pudota (drag-and-drop) logiikkajärjestelmää. Editorin logiikkajärjestelmä perustuu peliobjekteille määriteltäviin käytöskomentoihin sekä tapahtumapohjaiseen kulkukaavaan. Raahaa-ja-pudota-tyyppinen editori mahdollistaa pelinluomisen myös kehittäjille, joilla ei ole ohjelmointitaustaa. (Scirra, n.d.)

Valmiiden käytöskomentojen lisäksi, Construct 2 tukee lisäosaa, minkä avulla on mahdollista tuottaa omia käytöskomentoja JavaScript ohjelmointikielellä. Tämä mahdollistaa monipuolisempien käytösten ohjelmoimisen itse ja tuo siten paljon uusia mahdollisuuksia editorin käyttäjälle.

Editorin valmiit käytöskomennot ja raahaa ja pudota mekaniikka tekevät Construct2:sta myös hyvän testaustyökalun sekä nopean keinon luoda prototyyppejä. Peliään pääsee testaamaan napin painalluksella suoraa selainikkunassa ja uuden Preview Over Wifi -ominaisuutensa ansiosta myös muut samassa langattomassa verkossa olevat tietokoneet tai mobiililaitteet saavat esikatseluversion käyttöön helposti. Valitettavasti Preview Over Wifi on yksi niistä monista ominaisuuksista, jotka eivät kuulu Construct 2:n ilmaisversioon. (Garner, 2015; Scirra Instant Preview, n.d.)

Yksi Construct 2:n hyvistä puolista on sen tuki monelle eri alustalle. Peli kääntyy suoraan selaimille, mobiiliin sekä tietokoneille. Construct 2 tukee tällä hetkellä vain 2D-grafiikoita, mikä rajoittaa käyttäjäkuntaa nykypäivänä huomattavasti. Kevyiden ja yksinkertaisten mobiilipelien luomiseen se riittää, mutta monimutkaisempien projektien kohdalla peleistä tulee liian raskaita mobiililaitteella ajettaviksi. (O'Flanagan, n.d.)

Erilaisia lisenssivaihtoehtoja Construct 2:ssa on kolme. Free Edition on nimensä mukaan ilmainen, mutta siinä ei ole läheskään kaikkia ominaisuuksia ja perusominaisuuksienkin käyttöä on rajoitettu. Ohjelmiston kokeiluun ja pienien omaksi iloksi tehtävien projektien toteuttamiseen se riittää kuitenkin hyvin. Kotikäyttöön tarkoitettu Personal Edition poistaa kaikki rajoitukset ja tuo lisää ominaisuuksia käyttöön. Personal Editionin hinta on tällä hetkellä 99,99 € ja se tuo käyttöön kaikki ohjelman sisältävät toiminnot ja ominaisuudet. Kolmas versio on nimeltään Business License ja se on tarkoitettu niille, jotka tienaavat Construct 2 -ohjelmistolla tekemällä pelillään yli 5000 $ ja ovat osa organisaatiota joka tavoittelee voittoa. (Scirra Store, n.d.)

3.1.3 Unreal Engine

Epic Gamesin tuottama Unreal Engine on yksi pelimoottorimaailman vanhoista konkareista. Alun perin Unreal Engine kehitettiin Unreal-nimistä tietokonepeliä varten, minkä jälkeen se on ollut käytössä monissa, varsinkin FPS -peleissä (First Person Shooter / Ensimmäisen persoonan ammuntapeli). Vaikka Unreal on lähinnä FPS -peleihin kehitetty, on se uusilla versioillaan saanut jalansijaa myös mobiilipelimarkkinoilta. (Wikipedia Unreal, n.d.)

Unreal Engine on tehokas pelimoottori ja se on ollut pelinkehittäjien sydämissä jo pitkään. Unreal Enginen mukana tuleva lähdekoodi on yksi niistä syistä, miksi Unreal Engine on niin suosittu. Lähdekoodin avoimuuden salliman omien lisäosien ohjelmoimisen lisäksi, Epic Games julkaisi vuonna 2014 oman kauppapaikkansa, jossa kehittäjät voivat jakaa ja myydä tuottamaansa sisältöä. (Wikipedia Unreal Engine, n.d.)

Yksi Unreal Enginen myyntivalteista, on sen erittäin edistyneet grafiikat. Muihin pelimoottoreihin verratessa, Unreal Engine tarjoaa upean visuaalisen ulkoasun usealle eri alustalle. Tällä hetkellä Unreal Engine tukee useimpia kehittyneitä alustoja, niin konsoleita kuin mobiililaitteitakin.

Mobiilikehitykseen Unreal Engine tarjoaa nopean aloitusoppaan (Quick Start Guide) Android- ja iOS -alustojen kehittäjille. Tämän lisäksi Unrealin omasta dokumentaatiosta löytyy muun muassa ohjeet mobiilia varten optimoimiseen, grafiikkaan sekä mainonnan implementoimiseen. Unreal Engine on kohtalaisen nuori mobiilikehitysrintamalla, mutta kehitystä tapahtuu jatkuvasti. (Unreal Engine Mobile Game Development, n.d.)

Vuonna 2009 Epic Games julkaisi ilmaisen Unreal Development Kit -nimeä kantavan version Unreal Engine 3:n SDK:sta (Software Development Kit), mikä sai useat kehittäjät innostumaan kyseisen kehitystyökalun kokeilusta. Tuohon asti kehittäjien oli mahdollista tuottaa sisältöä, muttei julkaista sitä ilman kallista lisensointia. (Wikipedia Unreal Engine, n.d.)

Unreal Engine 4 avattiin yleisölle vuoden 2014 alkupuoliskolla, 19 $ kuukausihintaan. Vuoden 2015 Maaliskuussa Epic Gamesin perustaja Tim Sweeney ilmoitti Unreal Enginen uudesta suunnasta, mikä teki Unreal Engine 4:stä ilmaisen kaikille käyttäjille. Ilmaista versiota saa käyttää myös kaupallisiin julkaisuihin, mutta tällöin on maksettava 5 % tekijänoikeusmaksuja, kaikesta 3000 $ ylittävästä tienestistä per tuote per vuosineljännes. (Sweeney, 2015.)

3.1.4 Defold

Defold-pelinkehitysalusta edustaa pelinkehityksen uusia tuulia. Ruotsista lähtöisin oleva King-pelistudion uutinen Defold-pelinkehitysalustan vapauttamisesta, aiheutti vuoden 2016 alkupuolella paljon keskustelua mobiilikehittäjien keskuudessa. Täysin ilmaiseksi luvattu Defold, ei siis sisällä minkäänlaisia aloitus, kuukausi tai tekijänoikeusmaksuja. (Defold Terms of Service, n.d.)

Defold tarjoaa helpon alustan aloittaa pelinkehitys ja mainostaa itseään luovuuden vapauttamisella tekniikan liiasta monimutkaisuudesta. Kehitysalustan omilta sivuilta löytyy useita alkuun pääsemistä helpottavia tutoriaaleja sekä kattava manuaali kehittäjien käyttöön. Defold-kehittäjille on myös tarjolla oma foorumi, jossa Defoldin käyttäjät voivat jakaa ajatuksiaan ja mahdollisesti löytää ongelmiinsa ratkaisuja. (Defold Learn, n.d.)

Defoldin ohjelmointikielenä toimii Lua, jonka valitsemisen syiksi kehittäjät kertovat sen oppimisen helppouden sekä puhtaan ohjelmointikoodin. Defold tarjoaa myös useita valmiita ohjelmakoodin osia, joilla omien pelien toteuttaminen helpottuu. Defoldin avulla on myös mahdollista seurata tuotteensa pelaajien tilastoja, sisäänrakennetun analytiikkajärjestelmän avulla. (Defold, n.d.)

3.1.5 Päätös kehitysalustasta

Kehitysalustaksi päädyin valitsemaan Unityn sen laajan käyttäjäverkoston takia. Ohjeiden ja ongelmatilanteissa ratkaisujen helppo löytäminen oli minulle tärkeää, koska se nopeuttaisi projektin etenemistä huomattavasti. Unity on myös laajasti käytössä alan yrityksissä, mikä lisäsi mielenkiintoani sitä kohtaan jatkoa ajatellen.

3.2 Mobiilialustan valinta

Mobiilialustaksi valitsin Googlen omistaman Android-käyttöjärjestelmän. Android on minulle mobiilikäyttöjärjestelmistä tutuin, koska olen kyseisiä laitteita omistanut jo vuodesta 2010. Omistan useamman Android-laitteen, minkä vuoksi minun ei tarvitse käyttää virtuaalista Android-emulaattoria, vaan voin testata työtäni omilla fyysisillä laitteillani. Useamman eritasoisen laitteen omistaminen myös mahdollistaa pelin laajemman testauksen.

4 UNITY

Unity päivittyy monta kertaa vuodessa minor (pienemmin) päivityksin ja uusin patch (muutostiedosto) lisäyksin. Uusin major (suurempi) päivitys julkaistiin vuoden 2015 maaliskuussa, jolloin Unityn versio 5 näki päivänvalon. Jokainen versio tuo mukanaan jotain uutta, on se sitten lisää ominaisuuksia tai vanhojen ominaisuuksien korjauksia. Kirjoitushetkellä uusin versio on 5.3.5 ja tämän opinnäytetyön projekti aloitettiin käyttäen versiota 5.1.0. Projektin viimeiset muutokset on tehty versiolla 5.3.4f1. (Unity Download Archive, n.d.)

4.1 Unityn käyttöönotto ja perusteet

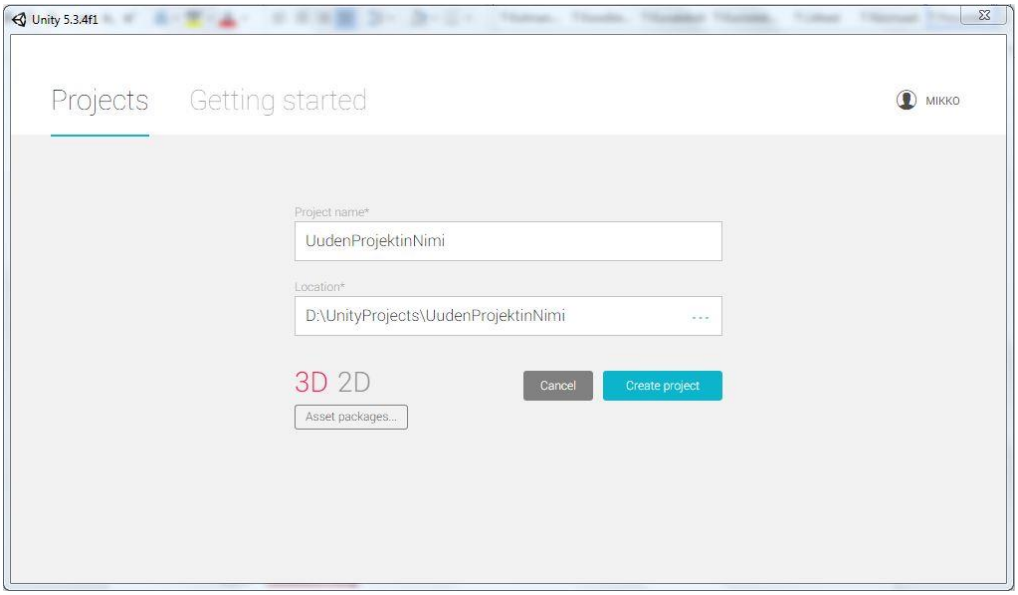
Tässä luvussa käydään läpi Unityn käyttöönottoon tarvittavat askeleet. Tutustutaan myös käyttöliittymään ja uuden projektin aloittamiseen.

4.1.1 Asennus ja uuden projektin luominen

Unityn asennus onnistuu helposti hakemalla ensin Unityn omilta verkkosivuilta haluamansa versio ja sen jälkeen ajamalla ladattavan UnityDownloadAssistant-tiedoston. Tämä lataa Unityn koneelle ja aloittaa asennuksen. Ruudussa näkyviä ohjeita seuraamalla, asennus onnistuu vaivattomasti. Asennuksen tultua päätökseen, luodaan käyttäjälle tili, jolla sitten kirjaudutaan Unityyn. Tilin luominen onnistuu nopeasti ja tämän jälkeen onkin Unity valmis käyttöön. (Unity Get Unity, n.d.; Unity Manual Downloading and Installing Unity, n.d.)

Uuden projektin luominen tapahtuu painamalla Projects-näkymästä New-painiketta. Tämä avaa uuden projektin luomisvalikon, joka sisältää kuvassa 4 näkyvät kohdat. Täytetään kenttiin haluttu projektin nimi ja sijainti, jonne projektin tiedostot halutaan tietokoneella sijoittaa. Tässä vaiheessa on myös mahdollista valita onko projekti 2D- vai 3D-pohjainen, sekä mahdollisesti

ladata projektiin valmiiksi joitakin resursseja. Tämä tapahtuu painamalla ikkunassa näkyvää Asset packages… -painiketta.



Kuva 4. Uuden projektin aloitusikkuna Unityssä.

4.1.2 Käyttöliittymä

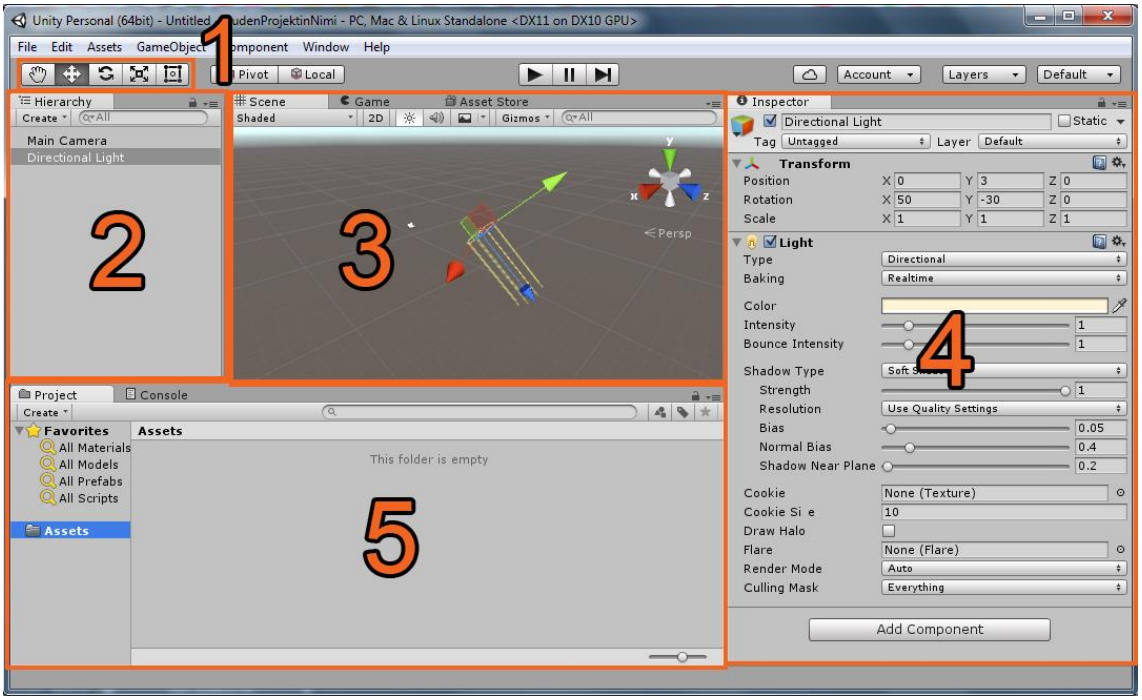
Unityn käyttöliittymä on modulaarinen ja täten muokattavissa käyttäjän omien tarpeiden mukaiseksi. Oletuksena käyttöliittymä jakautuu osiin kuvan 5 osoittamalla tavalla. Vasemmassa ylänurkassa sijaitsevat perustyökalut (kohta 1), joilla objekteja hallitaan ja liikutellaan. Tämän alapuolella Hierarchy (kohta 2) listaa kaikki ne objektit, jotka kyseiseen sceneen eli kohtaukseen on ladattuna. Listasta valitsemalla objekti aktivoituu, jolloin se on liikuteltavissa scene-näkymässä (kohta 3).

Scene-näkymässä voidaan sijoittaa objektit halutuille paikoilleen, jos ei sitä haluta tehdä antamalla tarkat arvot Inspector-näkymässä (kohta 4). Inspector-näkymässä näkyvät kaikki valittuna olevan objektin määritykset. Mallikuvassa näkyvissä esimerkkinä Directional Light -peliobjektin määritykset. Kaikki Inspector-näkymässä tehdyt säädöt tallentuvat objekteihin.

Kuvan alaosassa näkyvässä Project-näkymässä (kohta 5) näkyvät kaikki kyseiseen projektiin ladatut resurssit (Asset) eli muun muassa grafiikat, scriptit sekä scenet. Toisena vaihtoehtona alaosan ikkunalle on Console eli konsoli, johon esimerkiksi virheilmoitukset tulevat.

Scene-ikkunan vaihtoehtoina tarjotaan Game ja Asset Store -ikkunoita. Game-ikkunan näkymä on sama kuin tulevan pelin näkymä, ja painaessasi ylhäällä olevaa Play-symbolia, aukeaa se automaattisesti. Pelinäkymässä on siis näkyvissä se, miltä lopputulos näyttää. Oletuksena pelinäkymä on lähtöisin Main Camera -objektista, jonka näkymää peli siis seuraa.

Asset Store -ikkuna on nopea tie Unityn tarjoamaan Asset Storeen, josta on mahdollista ladata monenlaista hyödyllistä aineoista, kuten grafiikkaa ja valmiita ohjelmakoodeja erilaisiin tarkoituksiin.



Kuva 5. Unityn käyttöliittymä. Kuvaan merkittynä myös kohdat 1-5.

4.1.3 MonoDevelop

MonoDevelop on Unityn mukana toimitettava ohjelmointiympäristö eli IDE (integrated development environment). Unity ja MonoDevelop toimivat yhteen saumattomasti, mikä tekee MonoDevelopin käytöstä Unityn kanssa helppoa. MonoDevelop myös asentuu Unityn kanssa samasta paketista, mikä takaa sen, että jokaisella käyttäjällä on jonkinlainen alusta ohjelmakoodin editoimiseen, ilman että sellainen täytyy erikseen asentaa. Unity tukee muitakin ohjelmointiympäristöjä, mutta suosittelee käyttämään MonoDevelopia.

4.1.4 Scene

Scene tarkoittaa Unityssä yhtä aikaa ladattujen peliobjektien tilaa. Yksi scene on nimensä mukaisesti yksi kohtaus. Tämä tarkoittaa sitä, että kaikki mitä kyseisen kohtauksen aikana tarvitaan käyttöön, on oltava kyseiseen kohtaukseen tallennettuna. Voidaan myös ajatella että jokainen scene, on oma tasonsa pelissä. Pelin sisältämät erilaiset valikot ovat yleensä myös oma scenensä.

Scenellä ei ole varsinaisesti määriteltyä maksimikokoa, mutta varsinkin mobiilikehityksessä on muistettava, ettei mobiililaitteen muisti ole rajaton. Yhden tason ollessa suuri, on se mahdollista jakaa useampaan sceneen.

Osiin jaettua sceneä käyttäessä pitää kuitenkin ottaa huomioon se, että siirryttäessä scenestä toiseen tulee uusi scene ladata käyttöön, mikä tarkoittaa mahdollista latausaikaa. (Unity Manual Scenes, n.d.)

4.1.5 GameObject ja Prefab

GameObject eli peliobjekti, tarkoittaa Unityssä kaikkia sceneen ladattavia pelin osia. Jokainen pelihahmo, valo, tausta tai vaikkapa kamera, ovat peliobjekteja. Peliobjektit eivät itsessään tee mitään, mutta niihin liitetyt komponentit herättävät ne henkiin. Komponenttien avulla peliobjekteista tehdään niin pelin hahmot, viholliset sekä kaikki muu mitä pelissä tapahtuu. (Unity Script Reference GameObject, n.d.)

Prefab on jonkinlaisen peliobjektin malli. Prefab luodaan kääntämällä haluttu peliobjekti prefabiksi, mikä tallentaa kaikki kyseisen peliobjektin tiedot mallitiedostoon sisältäen myös sen komponentit ja niiden arvot. Näiden mallien käyttö on helppo tapa esimerkiksi luoda samanlaista vihollista useampi, tai toteuttaa vaikka jokainen pelihahmon keräämä kolikko. Prefabeja voi hyödyntää hyvin monissa erilaisissa tehtävissä, eikä niiden käyttö rajoitu pelkästään pelimaailmassa sijaitseviin objekteihin. Prefabeja on mahdollista siirtää myös projektien välillä, mikä mahdollistaa myös niiden lataamisen Unityn Asset Storesta. Unity tarjoaa itse myös muutamia prefabeja kehittäjien käyttöön.

Prefabit voidaan ladata sceneen ohjelmakoodista, joka mahdollistaa niiden luomisen ja monistamisen siinä kohta kun on tarve. Prefabien käyttö, myös keventää pelien latausaikoja, sillä ilman prefabejä, jokainen peliobjekti olisi ladattava omana objektinaan sceneen jo valmiiksi. (Unity Manual Prefabs, n.d.)

4.1.6 Script

Script-tiedostot, ovat jollain Unityn tukemalla ohjelmointikielellä kirjoitettuja komentosarjoja. Ne liitetään yleensä peliobjektiin, jolloin peliobjektiin saadaan eloa. Lähes kaikki mitä pelin sisällä tapahtuu, on lähtöisin scripteistä. Suurin osa Unityn käyttäjistä käyttää ohjelmointikielenään C#:ia, mutta myös UnityScript (muistuttaa JavaScriptiä) sekä Boo ovat käytössä. (Unity Manual Creating and Using Scripts, n.d.)

Script-tiedostoilla luodaan peliobjekteille käytöksiä. Unity tarjoaa käyttäjilleen valmiin kirjaston, joka sisältää useita hyödyllisiä funktioita. Yksi tärkeimmistä Unityn tarjoamista luokista on MonoBehaviour. Tämä sisältää muun muassa Start()- ja Update() -funktiot. Näiden avulla on mahdollista määrittää toimintoja peliobjekteille, kun scene ensimmäistä kertaa ladataan (Start()-funktio) tai jokaisen ruudunpäivityksen yhteydessä (Update()-funktio). MonoBehaviour sisältää monia muitakin funktioita, kuten fysiikkamoottoriin yhdistetyt OnCollison-funktiot, jotka mahdollistavat pääsyn fysiikkamoottorin tietoihin objektien kosketuksista. (Unity Script Reference MonoBehaviour, n.d.)

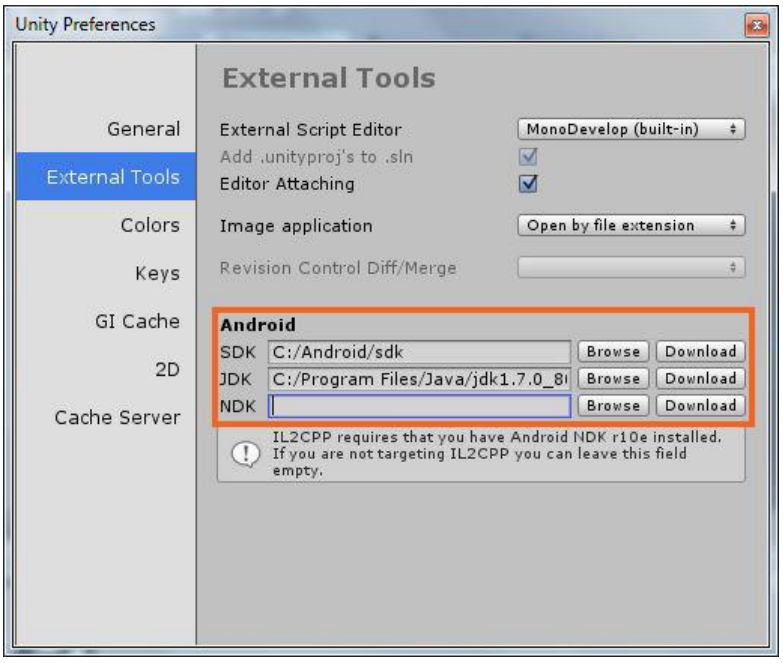
4.2 Unity mobiilikehitysalustana

Tässä luvussa käsitellään Unityn käyttöä mobiilikehitysalustana. Tutustutaan asennusprosessiin sekä muutamaan mobiilikehityksessä hyödylliseen työkaluun.

4.2.1 Android SDK:n asennus ja käyttöönotto

Unity tarjoaa Android-kehittäjälle nopean aloitusoppaan, joka käy läpi kehityksen aloittamisen perusvaiheet. Käytännössä tarvitsee vain asentaa Android Studio, jonka linkki löytyy Unityn oppaasta sekä itse ohjelman sisältä ja yhdistää se Unityyn. On myös mahdollista ladata pelkkä SDK (Software Development Kit), mutta Android Studio sisältää muutakin hyödyllistä, joten sen lataaminen kokonaisuudessaan kannattaa. (Unity Manual Android SDK Setup, n.d.)

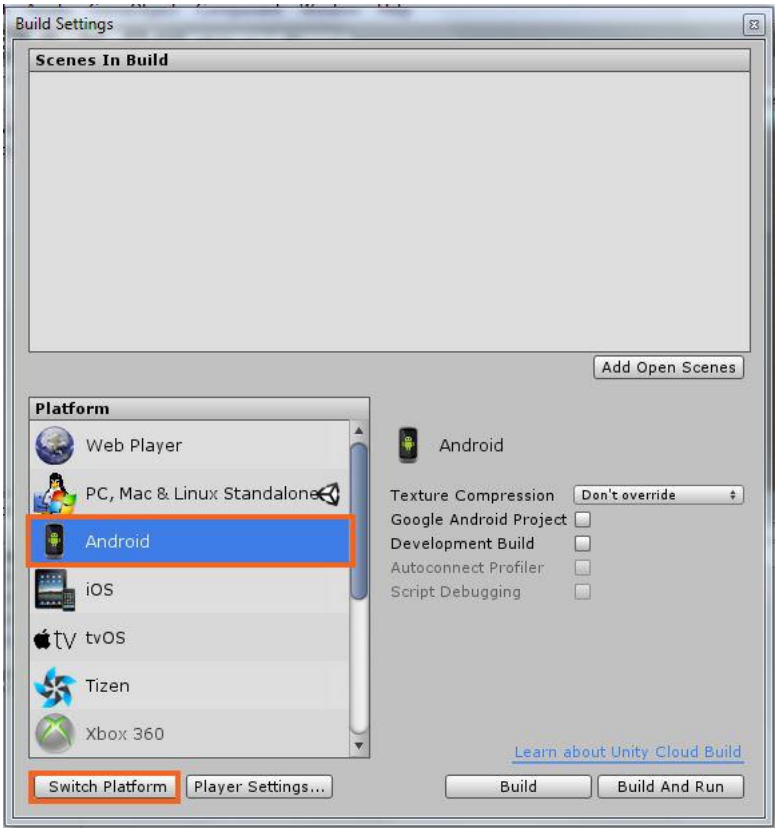
Toimivan Android Studion asennuksen yhdistäminen Unityyn onnistuu avaamalla Unityn Edit > Preferences… > External Tools, ja hakemalla Browse-painikkeella asennetun Android SDK:n sijainti, kuvan 6 mukaisesti. Android Studion asennuksen vielä puuttuessa, pääsee Download-painiketta painamalla ohjelmiston lataussivulle.



Kuva 6. Unityn External Tools eli ulkoiset työkalut valikko.

Android-studion asennuttua liitetään halutessaan Android-laite tietokoneeseen, jota onnistuneen yhdistämisen jälkeen voidaan käyttää testilaitteena. Ongelmatilanteita laitteen yhdistämisessä aiheuttavat joskus laitteiden ajurit, eli tietokone ei tunnista laitetta oikein, eikä siihen tästä syystä pääse Unityllä tai muilla kehitystyökaluilla käsiksi. Android Studiossa toimitetaan mukana ajurit joiden pitäisi toimia, mutta jos näin ei ole niin Unity kehottaa etsimään apua tarjoamaltaan Troubleshooting Android Development -sivulta. (Unity Manual Troubleshooting Android Development, n.d.)

Oletuksena kaikki luodut projektit ovat pöytätietokoneille kohdistettuja, mutta tämän vaihtaminen haluttuun kohdealustaan onnistuu helposti avaamalla File > Build Settings… valikko. Kuvan 7 mukaisen ikkunan auetessa klikataan aktiiviseksi Android-teksti ja painetaan Switch Platform -painiketta, jolloin kohdealustaksi muuttuu Android.

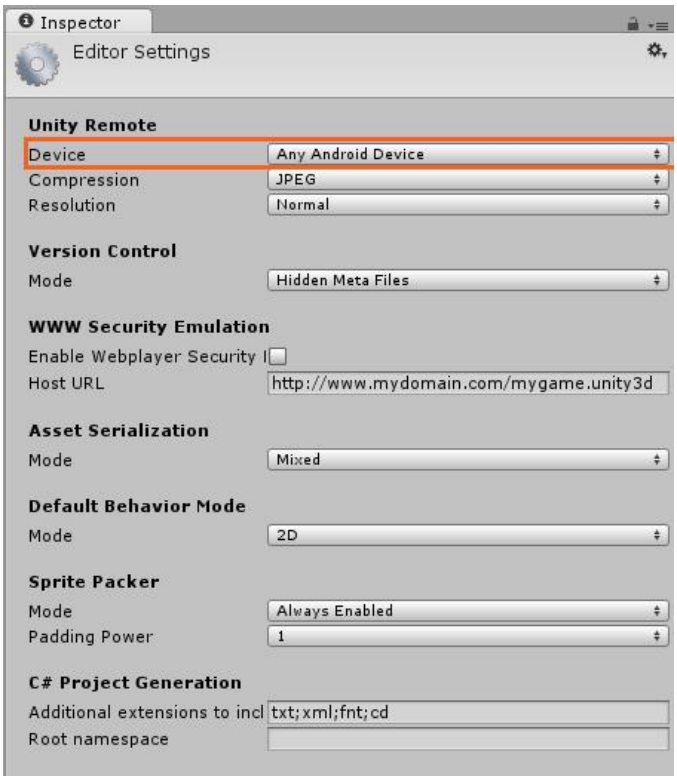


Kuva 7. Unityn Build Settings. Alustan vaihto tapahtuu valitsemalla jokin aktiiviseksi ja painamalla Switch Platform -painiketta.

4.2.2 Unity Remote

Unity Remote on Unityn tarjoama esikatselutyökalu, joka mahdollistaa mobiilikehittäjille oikealla laitteella testaamisen, luomatta aina uutta koontiversioita (build). Testilaitteeseen asennetaan Googlen Play Storesta löytyvä Unity Remote -ohjelma. Asennettu ohjelma avataan laitteessa ja laite kytketään tietokoneeseen. Toimiakseen oikein Unity tulee käynnistää vasta Unity Remoten ollessa käynnissä ja testilaitteen ollessa kytkettynä tietokoneeseen.

Unityssä laitteen aktivoiminen tapahtuu avaamalla Edit > Project Settings > Editor ja sieltä valitsemalla laite käyttöön kuvan 8 mukaisesti. Device-kohdan None vaihdetaan valintaan Any Android Device pudotusvalikosta, jolloin oikeinyhdistetty laite saa yhteyden Unityyn. Painamalla Play-symbolia Unityssä aukeaa peli nyt myös mobiililaitteen näytölle. Tämä mahdollistaa kosketuskontrollien testaamisen, mikä olisi muuten mahdotonta ilman erilistä kosketusnäyttöä.



Kuva 8. Unity Remoten käyttöönottoon tarvittavat säädöt.

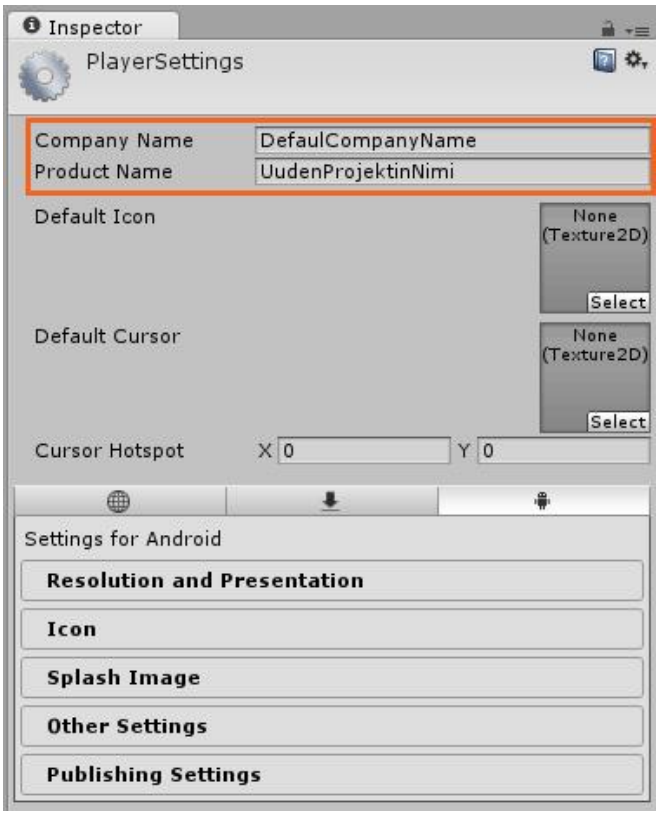
4.2.3 Canvas Scaler

Canvas Scaler on scripti joka skaalaa Unityn käyttöliittymäelementit eri kokoon valintojen mukaan. Skaalauksen pystyy määrittämään kolmella eri tavalla. Oletuksena valinta on Constant Pixel Size, joka pitää pikselit eli kuvapisteet samankokoisina, ruudun koosta riippumatta. Vaihtoehtoisesti Scale With Screen Size kasvattaa elementtejä sitä enemmän, mitä suurempi näyttö on käytössä. Kolmantena vaihtoehtona on tarjolla Constant Physical Size, jonka avulla elementit säilyttävät oman kokonsa näytön koosta ja resoluutiosta riippumatta. (Unity Manual Canvas Scaler, n.d.)

Varsinkin mobiilikehityksessä Canvas Scaler on hyödyllinen työkalu, koska laitteita on tuhansia erilaisia. Valitsemalla näytön koon mukaisen skaalauksen oikeilla asetuksilla, on mahdollista julkaista suurelle osalle mobiililaitteista ilman jokaiselle oman version toteuttamista.

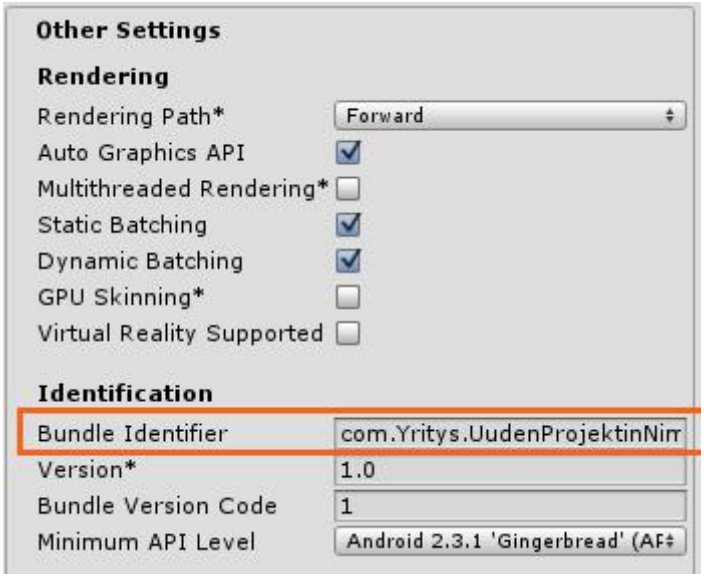
4.2.4 Projektin asetukset mobiilijulkaisua varten

Unity vaatii Android SDK:n kytkettynä olemisen lisäksi muutaman asetuksen, jotta se pystyy rakentamaan koontiversioita. Asetuksiin pääsee avaamalla Build Settings… -ikkunan ja klikkaamalla Player Settings -painiketta. Oikeaan laitaan Inspector-ikkunan paikalle aukeaa kuvan 9 mukainen valikko, jossa tarvittavat asetukset sijaitsevat. Ensimmäisessä paneelissa tulee antaa Company Name eli yrityksen nimi. Toisessa kentässä annetaan pelille tai tuotteelle nimi (Product Name).



Kuva 9. Player Settings valikon perusnäkymä.

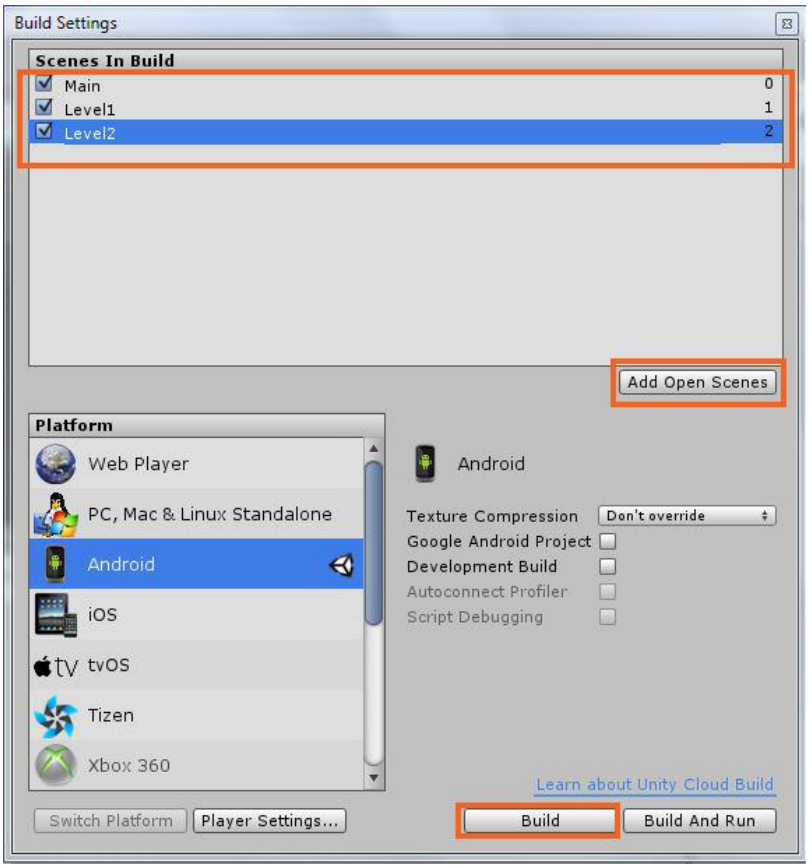
Näiden alapuolella on mahdollista säätää pelin pikakuvaketta ja monia muitakin tarpeellisia ominaisuuksia. Viimeinen Unityn vaatimista asetuksista löytyy kuvan 10 mukaisen Other Settings -välilehden alta kohdasta Bundle Identifier, johon tulee kirjoittaa oman projektinsa tunniste hyväksyttävässä muodossa. Unityn hyväksymä muoto on esimerkiksi com.YrityksenNimi.TuotteenNimi. Player Settings -asetusvalikko sisältää kattavan valikoiman android-julkaisemiseen hyödyllisiä ominaisuuksia, joihin on syytä tutustua omien tarpeiden mukaan tarkemmin.



Kuva 10. Player Setting -valikon Other Settings -välilehti.

4.2.5 Lopullisen tuotteen käyttöönotto ja julkaisu

Valmiin tuotteen käyttöönotto onnistuu rakentamalla apk-tiedosto ja ajamalla se mobiililaitteessa. Unityssä apk-tiedoston rakentaminen onnistuu menemällä Build Settings -asetuksiin ja lisäämällä halutut scenet Scenes in Build -listaan, kuvan 11 mukaisesti. Tämä onnistuu joko painamalla Add Open Scenes -painiketta, joka lisää avoinna olevat scenet listaan, tai hinaamalla halutut scenet listan päälle project-näkymästä.



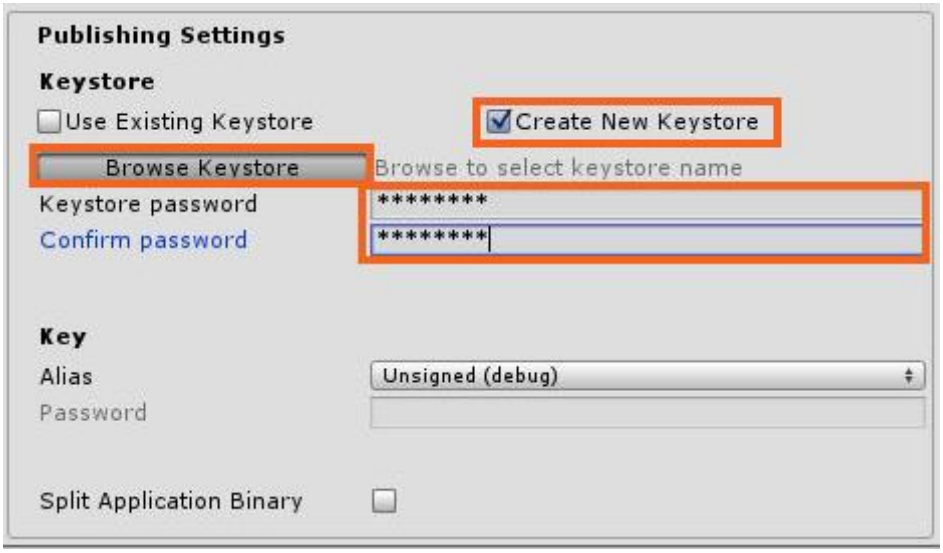
Kuva 11. Build Settings ja siihen lisätyt scenet.

Painiketta Build painamalla projekti kääntyy apk-tiedostoksi, jonka voi sitten asentaa android-laitteeseen. Build And Run -painiketta painamalla pelitiedosto myös ajaa itsensä yhdistetyssä laitteessa automaattisesti.

Toteuttamiaan android-projekteja on mahdollista julkaista monessa eri kauppapaikassa, mutta näistä ehdottomasti suosituin on Googlen omistama Play Store. Play Storeen pelejä ladatakseen tarvitsee kehittäjän avata Google Developer -tili, joka maksaa kirjoitushetkellä 25 $. Maksu tarvitsee tehdä vain kerran, eikä erillisistä julkaisuista peritä lisämaksuja. (Android Developers Get Started with Publishing, n.d.)

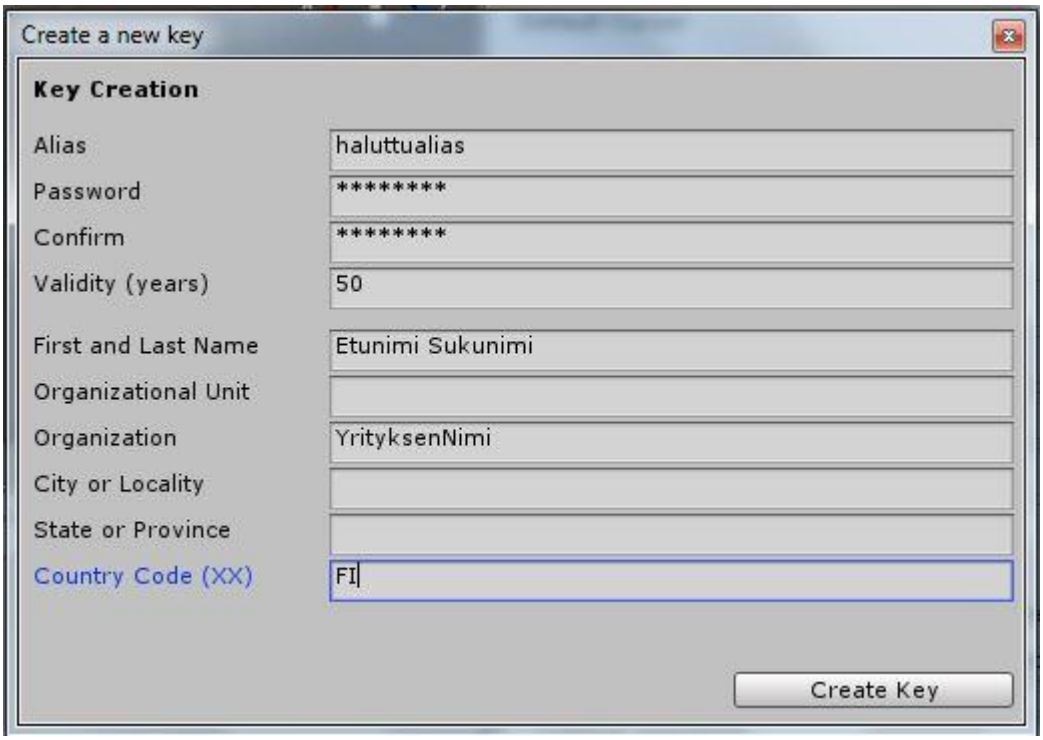
Googlen Play -kaupassa julkaisu vaatii apk-tiedoston allekirjoittamisen mikä on mahdollista toteuttaa Unityn sisällä tai jälkikäteen Android Studiolla. Unityn sisällä allekirjoittaminen onnistuu menemällä Player Settings -valikkoon ja sieltä alimpaan Publishing Settings -osaan.

Aluksi luodaan uusi Keystore, Create New Keystore kohtaa painamalla, kuvan 12 mukaisesti. Haetaan Browse Keystore -painikkeella Keystorelle haluttu tallennussijainti ja annetaan sille salasana.



Kuva 12. Publishing Settings Keystoren luomiseen.

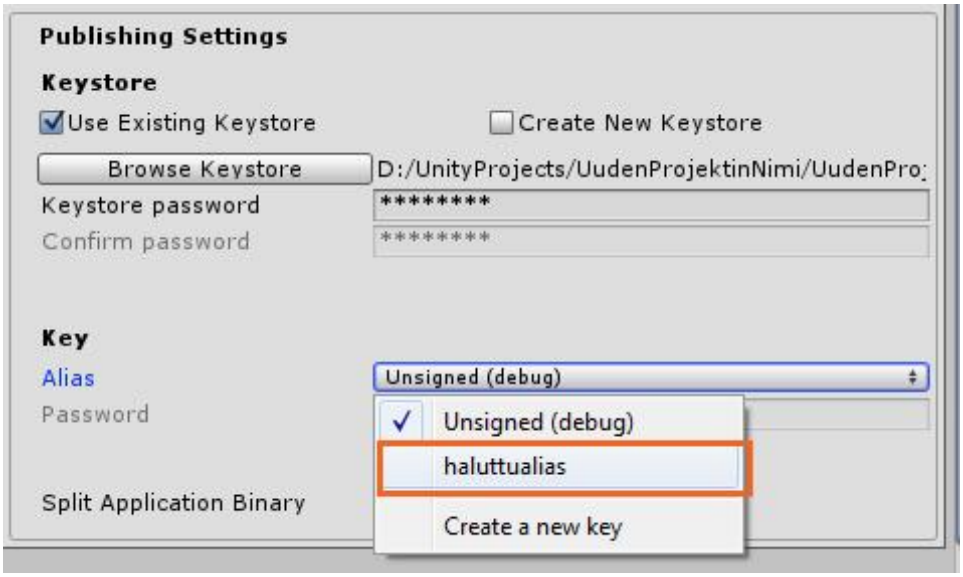
Tämän jälkeen valitaan Alias-kohdan tiputusvalikosta kohta Create a new key, joka avaa kuvan 13 mukaisen ikkunan tietojen täyttämistä varten. Täytetään halutut kohdat, mutta vähintään Alias, Password, Confirm sekä Validity -kohdat ja painetaan Create Key -painiketta.



Kuva 13. Create a new key -valikko, apk-tiedoston allekirjoittamista varten.

Haetaan Alias-kohdan tiputusvalikosta juuri luotu Alias, kuvan 14 mukaisesti ja täytetään sen alle sille määritelty salasana. Tämän jälkeen projekti

on valmis rakennettavaksi. Keystoren ja Keyn lisääminen projektiin allekirjoittaa sen määrittämälläsi tunnuksella, jolloin sitä ei voi päivittää kukaan, joka ei kyseistä tunnusta omista.



Kuva 14. Publishing Settings uuden Aliaksen luomisen jälkeen.

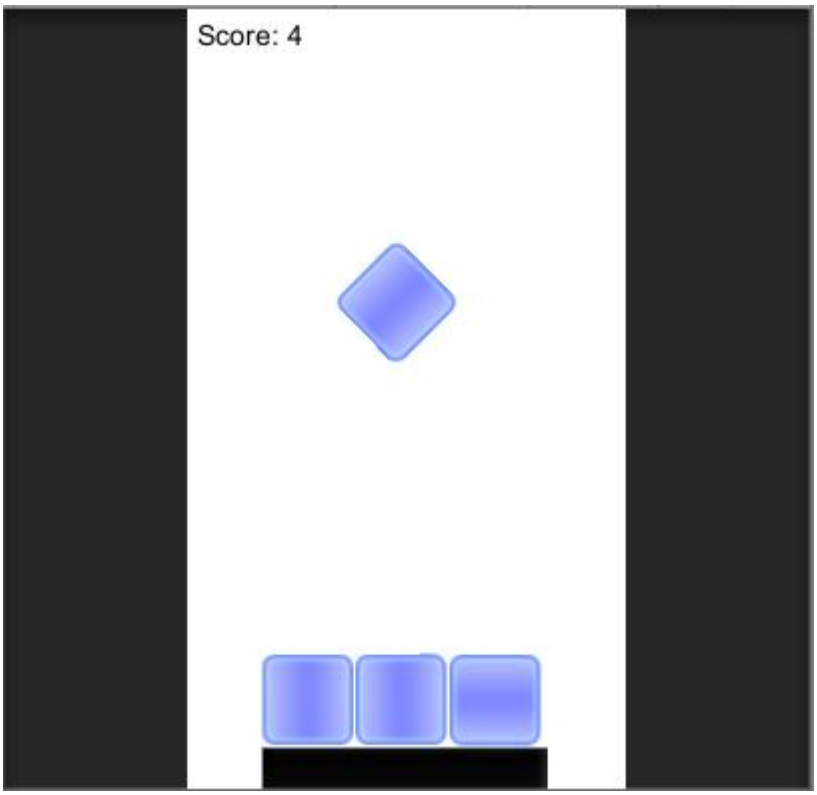
Valmiin apk-tiedoston voi nyt ladata Googlen Play -kauppaan Developer Consolen kautta. Googlen Play -kauppaan pelejä ladatessa on myös täytettävä erilaisia tietoja esimerkiksi siitä, onko sovellus maksullinen vai ilmainen sekä suositeltu ikäryhmä. Näiden tietojen ollessa täytetty, on peli valmis julkaistavaksi. Google tarjoaa kehittäjille Launch Checklist -sivun, josta löytyy paljon muutakin, ennen ja jälkeen julkaisun tehtävää. (Android Developers Launch Checklist, n.d.)

5 PELIN TOTEUTUS

Tässä luvussa käsitellään toteutetun pelin toimintaa ja rakennetta.

5.1 Pelin idea

Pelin ideana oli perinteinen laatikoiden tai muiden muotojen päällekkäin kasauspeli, kuvan 15 mukaisesti. Lähdin toteuttamaan peliä yhdistellen erilaisten tutoriaalien ohjeita. Halusin luoda pelin, jossa pelaaja pääsisi etenemään uusille tasoille vasta läpäistyään edellisen tason tavoitteen. Pisteitä saisi jokaisesta kenttään ilmestyneestä objektista ja kun tason läpäisyyn vaadittava määrä pisteitä oli saatu, avautuisi seuraava taso valikossa. Taso tulisi läpäistä ilman että yksikään objekti tippuu pelialustalta ennen kuin vaadittava määrä pisteitä olisi kasassa.



Kuva 15. Pelinäkymä.

5.2 Pelimekaniikan toteutus

Pelin perusmekaniikan suunnittelemisen aloitin seuraamalla erilaisia tutoriaaleja 2D-fysiikkapohjaisen pelin luomiseen. Päädyin pääsääntöisesti käyttämään Quill18-nimisen tutoriaalien tekijän 2d Physics Mobile Game -nimistä tutoriaalisarjaa. Pelissä toteutetaan muotoja ampuva peliobjekti ja määritetään, kuinka monta pistettä pelaajan tulee saada kullakin tasolla. (Quill18, n.d.)

5.2.1 Kontrollit

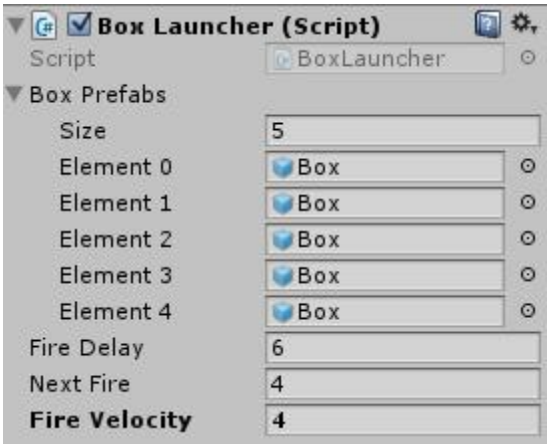
Kontrolleina toimisivat näytön kosketukset mobiililaitteella. Tämä oli mahdollista toteuttaa monella tapaa Unityssä, mutta päädyin käyttämään samoja ohjelmakoodeja kuin hiiren painikkeita käytettäessä. Unity osaa itse kääntää hiiren painallukset mobiililaitteille sopiviksi kosketuksiksi. Tämän lähestymistavan mahdollisti se, että en tarvinnut muita tietoja kuin kosketuksen sijainnin, enkä esimerkiksi kiihtyvyysanturin lukemia.

Pelaaja siis ottaa lentävän laatikon kiinni, sitä koskemalla ja pitämällä siitä kiinni oli tämän mahdollista liikuttaa objekti haluamalleen paikalle. Kosketukselle luotiin sijainti siihen kohtaan peliobjektia, mihin pelaaja koski. Laatikkoa hinatessa luotiin sen perään ruudullakin näkyvä linja (dragLine), joka kulki siitä pisteestä mistä pelaaja otti kiinni objektissa, siihen pisteeseen missä pelaajan sormi tällä hetkellä on.

Objektin liikuttelua on rajoitettu siihen lisättyä fysiikkaa säätämällä. Jokaisella pelissä kasattavalla muodolla on Rigidbody2D, jonka Linear drag ja Angular drag -arvoja säätämällä, ne saadaan liikkuvaan hieman kontrolloidummin. Linear drag vaikuttaa objektiin kohdistuvaan kitkaan sijainnista riippuen ja angular drag vaikuttaa sen pyörimisliikkeeseen kitkan vaikutuksen alaisena. (Unity Manual Rigidbody2D, n.d.)

5.2.2 BoxLauncher ja DeathTrigger

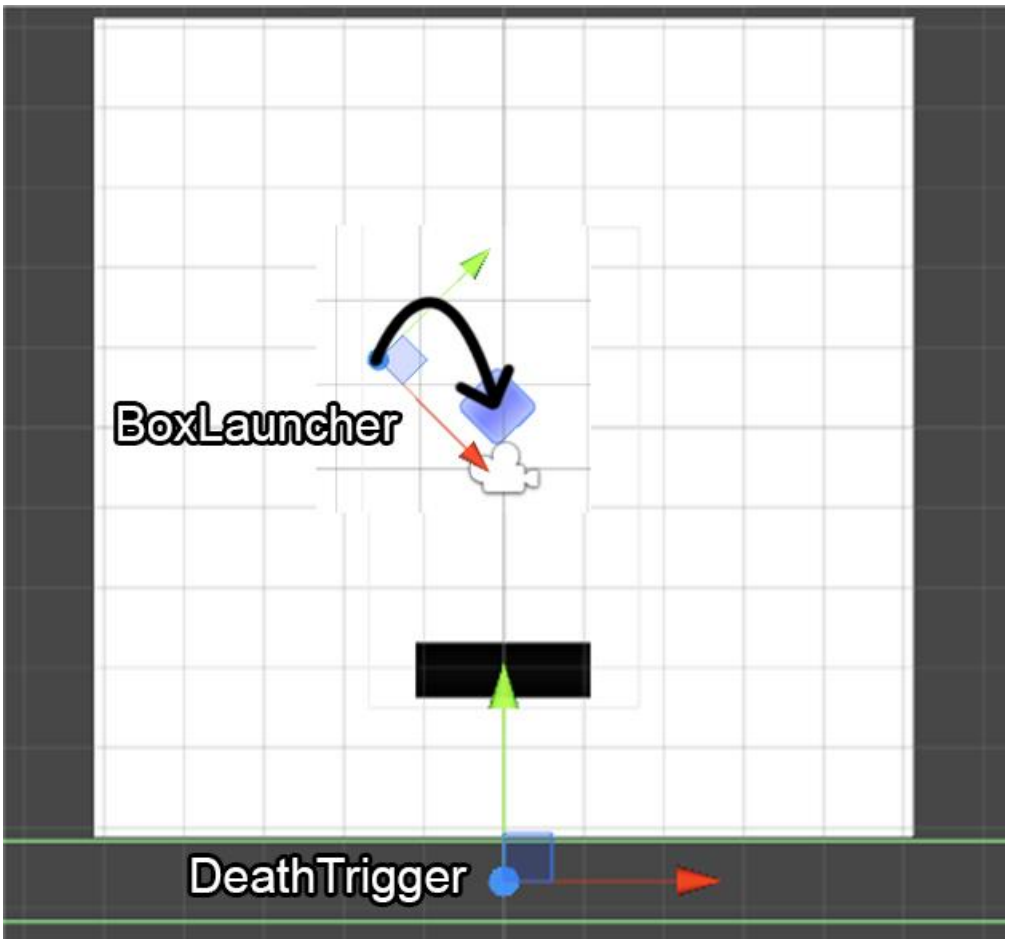
Halusin että peliobjektit luodaan peliin tasaisin väliajoin automaattisesti, ettei pelaajalla ole liikaa aikaa rakennella torniaan. Tämä toteutui ohjelmoimalla BoxLauncher, eli laatikon laukaisija, johon oli mahdollista määrittää muitakin muotoja. BoxLauncher-peliobjektin kulmaa säätämällä oli mahdollista säätää lähtevän ammuksen kulmaa. BoxLauncheriin liitetyn scriptin ansiosta oli mahdollista myös määrittää ammuksen lähtönopeus sekä intervalli, jolla ammukset lähtisivät. Nämä ovat kuvan 16 mukaisesti säädettävissä myös Inspector-näkymässä. Muuttamalla Element-kenttään valittu Box-prefab toiseen prefabiin, on mahdollista ampua muitakin kuin kyseisiä Box-prefabejä.



Kuva 16. BoxLauncher-scriptin määritykset Inspector-näkymässä.

BoxLauncher-scriptissä monistetaan aiemmin luotu Box-prefab, säädetyn intervallin välein. Tämä siis luo kopion aiemmin määritetystä muodosta ja ampuu sen määritetyllä lähtönopeudella. BoxLauncher-scripti myös laski pisteet lisäämällä score-muuttujaan aina yhden, kun ammus oli ammuttu.

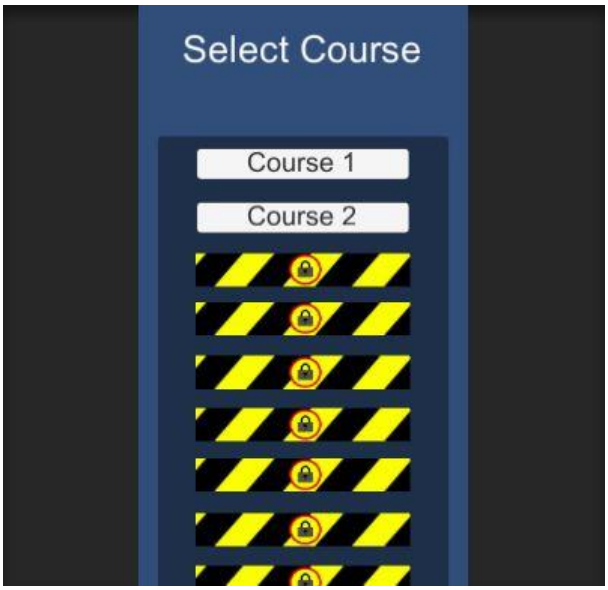
Peliobjektien laukaiseminen loppuu, jos jokin näistä objekteista on koskettuksissa DeathTrigger-peliobjektiin, joka sijaitsee pelialustan alla, kuten kuvassa 17 näkyy. DeatTrigger-peliobjekti on Collider, joka toimii Triggerinä eli kun jokin määritykset täyttävä objekti osuu alueeseen, ilmoittaa se scriptille näin käyneen. Triggerin aktivoituessa, DeathTrigger-scriptistä löytyvä hasLost-muuttuja vaihtaa arvoaan, ja tämä lopettaa BoxLauncherin toiminnan ja tulostaa näytöllä tekstin Game Over ja Restart-painikkeen, jota painamalla pelaaja pääsee yrittämään uudestaan.



Kuva 17. BoxLauncher sekä DeathTrigger -peliobjektit havainnollistettuna.

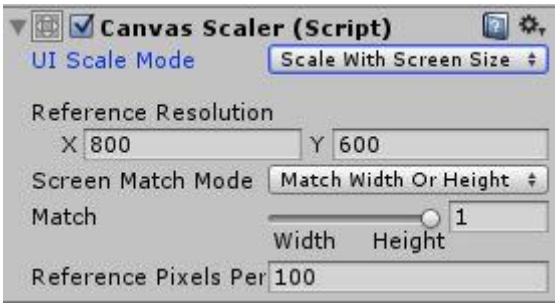
5.3 Käyttöliittymän suunnittelu

Valikon käyttöliittymään halusin mahdollisuuden lukita läpäisemättömät pelin tasot ja pelattavan tason valinnan. Käyttöliittymän toteutin käyttämällä Unityn omaa GUI:tä (Graphical user interface) eli graafista käyttöliittymää, kuvan 18 mukaisesti. GUI mahdollisti interaktiivisten painikkeiden (Button) luomisen sekä kaiken muun haluamaani valikkojen ulkoasuun tarvittavan.



Kuva 18. Päävalikon näkymä kun ensimmäinen taso on läpäisty.

Päävalikossa käytin hyödyksi CanvasScaler-scriptiä, jonka avulla onnistuin skaalaamaan valikon elementit oikean kokoisiksi käytettävästä laitteesta riippumatta. Säädin CanvasScaler-scriptin kuvan 19 mukaisesti, muuttamaan elementtien koon näytön koon mukaan eli valinnalla Scale With Screen Size.



Kuva 19. Canvas Scaler -scriptin asetukset.

Pelin aikana ruudussa ei näy pelin itsensä lisäksi muuta kuin vasemmassa ylänurkassa saavutettu pistemäärä. Tämän toteutin ohjelmakoodissa luomalla päivittyvän tekstin, jonka ensimmäinen osa ”Score:” pysyy samana ja sen perään tulostettava score-muuttujan arvo päivittyy pelaajan pistemäärän mukaan.

5.3.1 Tasojen läpäisy

Määrittelin jokaisen pelin tason nimet niin, että ne sisältävät ns. maailman ja tason järjestysnumerot. Maailma tässä tapauksessa tarkoittaa useamman tason kokonaisuutta. Tässä projektissa en kuitenkaan tarvinnut kuin yhtä maailmaa, mutta määrittelemällä tasojen nimet näin, mahdollistin jatkossa niiden helpomman luomisen. Tasojen nimet ovat siis muotoa LevelX.Y, jossa X on maailman järjestysnumero ja Y on tason vastaava arvo. (Horakeri, 2014.)

Tasojen nimiä hyväksi käyttäen loin ScoreManager-scriptiin (kuva 20) goalScore-muuttujan, joka määrittää kuinka monta pistettä kunkin tason läpäisemiseen vaaditaan. Kun määritelty pistemäärä on saavutettu, aukeaa ruudulle teksti ” Passed!” ja mahdollisuus siirtyä napilla takaisin päävalikkoon valitsemaan seuraava taso.

```
15    public int goalScore = 0;
16
17    void Start () {
18        //tallenna nykyisen tason nimi
19        currentLevel = Application.loadedLevelName;
20
21        Debug.Log(currentLevel);
22
23        if (currentLevel == "Level1.1")
24            goalScore = 5;
25        if (currentLevel == "Level1.2")
26            goalScore = 10;
27        if (currentLevel == "Level1.3")
28            goalScore = 15;
```

Kuva 20. Kunkin tason tavoitteellisen pistemäärän määrittävä, goalScore-muuttuja ScoreManager-scriptissä.

5.3.2 Tasojen lukitus ja avaus

Tason läpäistyä asetetaan ScoreManager-scriptissä PlayerPrefsille arvo, josta LevelSelectScript tietää kyseisen tason olevan läpäisty siirryttäessä valikkoon, ja määrittää sen sitten ei lukituksi. Valikossa LevelSelectScript käy läpi kaikki tasot ja avaa sitten ne tasot, joiden läpäisytiedon se löytää LockedLevel-scriptin avulla. Tämän jälkeen se tulostaa näytölle lukon kuvat kaikkien niiden tasojen eteen, jotka ovat vielä lukittuja ja estää niiden painikkeen toiminnan. (Unity Script Reference PlayerPrefs, n.d.)

5.4 Kameran liike

Pelin toimintaan vaadittiin vielä muutama avustava scripti. Yksi näistä on kameran liikettä ohjaava CameraMover-scripti ja sen apuna CollisionHandler-scripti. CollisionHandler-scripti tunnistaa törmäykset ja antaa CameraMover-scriptille tiedon kuinka paljon sen pitää liikkua y-akselilla ylöspäin, jotta näkymä pysyy pelaajan mukana. Lisäsin CameraMover-scriptiin Lerp-komennon, joka pehmentää liikkumista, eikä kameran liike näytä niin katkonaiselta. (Unity Script ReferenceMatf.Lerp, n.d.)

6 LOPPUSANAT JA YHTEENVETO

Mobiilipelin toteuttamisessa on useita eli aspekteja, joita asiaan tutustumaton ei välttämättä tiedosta. Mobiililaitteille pelinkehittäminen tuo mukanaan niin rajoituksia kuin vapauksiakin. Pääosin kosketusnäytöllä toteutettavat kontrollit on rajoitettava muutamaan eri painikkeeseen tai eleeseen, muuten

näytön alasta peittyy liian suuri osuus, eikä pelaaja enää näe tarpeeksi. Uusien ohjaustekniikoiden, kuten kiihtyvyysanturien käyttö, mahdollistaa kuitenkin monipuolisemman pelikokemuksen.

Tavoitteena oli toteuttaa toimiva mobiilipelidemo ja pelin toteutus onnistui kutakuinkin niin hyvin, kuin olin toivonutkin. Ensimmäiseksi peliprojektikseni olisin kuitenkin voinut valita vielä yksinkertaisemman idean, jotta minulla olisi ollut enemmän aikaa viimeistellä peliä ja tutustua tarkemmin julkaisuvaiheeseen. Tällä hetkellä projektinani ollut mobiilipeli, ei ole siinä tilassa, että sen voisi julkaista. Pelattavuus, valikot ja muu tavoitteeksi asettamani toimii, mutta esimerkiksi grafiikoiden osalta en päässyt toteuttamaan peliä niin paljon kuin olisin ehkä halunnut.

Tätä projektia toteutettaessa opin paljon siitä, mitä kaikkea mobiilipelinkehittäminen pitää sisällään. Kuinka paljon aikaa jokainen eri vaihe vie, kun yksikin ongelma ilmenee tai kun jonkin osan toiminta ei ollutkaan ajateltu loppuun asti.

Opin työtä tehdessäni myös huomattavan määrän ohjelmoinnin rakenteesta, jota en ollut aikaisemmin ohjelmointiopinnoissani ymmärtänyt. Monelle peliprojektin tekeminen voisikin olla helpompi tapa ymmärtää ohjelmoinnin perusteita, varsinkin jos apuna käytettäisiin jotain valmista kehitysympäristöä. Tämä mahdollistaisi pienimuotoisen tekemisen, ennen kuin ymmärrys kokonaisuudesta on saavutettu ja täten voisi auttaa ymmärtämään joitain rakenteita helpommin.

Projektia olisi voinut jatkaa viemällä peli julkaisukuntoon asti muun muassa grafiikoiden osalta, jolloin pelin tuntuma olisi parantunut huomattavasti. Visuaalinen ilme peleissä on nykypäivänä lähes yhtä tärkeää kuin itse pelattavuuskin. Muutamia pieniä yksityiskohtia jäi myös toteuttamatta oman osaamisen ja ajallisen suunnittelun heikkouden takia. Erilaiset tasoideat ja pisteytystavat olisivat olleet myös kiinnostavia toteuttaa.

Jatkossa aion hyödyntää oppimiani tekniikoita uusien pelien kehittämiseen ja mahdollisesti myös tämän projektin viimeistelyyn. Toivonkin saavani mahdollisuuden paneutua tähän ja muihin peliprojekteihin jatkossakin. Ohjelmoimisen rakenteen parempi ymmärrys, ei myöskään varmasti mene hukkaan työmarkkinoilla.

LÄHTEET

Android Developers: Get Started with Publishing (n.d.), retrieved on 29.05.2016, from
https://developer.android.com/distribute/googleplay/start.html

Android Developers: Launch Checklist (n.d.), retrieved on 27.05.2016, from
https://developer.android.com/distribute/tools/launch-checklist.html

Crawley, Daniel: 5 Lessons in Mobile VR Development from One of Gear VR's Early Hitmakers (2016), retrieved on 20.05.2016, from
https://www.chartboost.com/blog/2016/04/5-lessons-in-mobile-vr-development-from-one-of-gear-vrs-early-hitmakers/

Defold Learn (n.d.), retrieved on 31.05.2016, from
http://www.defold.com/learn/

Defold Product (n.d.), retrieved on 31.05.2016, from
http://www.defold.com/defold/

Defold Terms of Service (n.d.), retrieved on 31.05.2016, from
http://www.defold.com/about-terms/

Fingersoft: Fast Like a Fox (2015), retrieved on 19.05.2016, from
http://fingersoft.net/games/fast-like-a-fox/

Garner, Caleb: Construct 2: The Ultimate 2D Prototyping Tool (2015), retrieved on 25.05.2016, from
https://www.playcrafting.com/construct-2-the-ultimate-2d-prototyping-tool/

Horakeri, Sujit: Create A Level Select Scroll Menu - Unity 4.6. (2014), retrieved on 26.05.2016, from
http://www.thegamecontriver.com/2014/09/create-level-select-scroll-menu-unity-46.html

Kuorikoski, Juho: Finnish Video Games: A History and Catalog (2015) Yhdysvallat, Pohjois-Carolina, McFarland, ISBN: 9781476621197

Mobile Industry Review: How will virtual reality influence mobile gaming? (2016), retrieved on 20.05.2016, from
http://www.mobileindustryreview.com/2016/05/will-virtual-reality-influence-mobile-gaming.html

Newzoo: THE GLOBAL GAMES MARKET REACHES $99.6 BILLION IN 2016, MOBILE GENERATING 37% (2016), retrieved on 31.05.2016, from
https://newzoo.com/insights/articles/global-games-market-reaches-99-6-billion-2016-mobile-generating-37/

O'Flanagan, Jamie: Game Engine Analysis and Comparison (n.d.), retrieved on 25.05.2016, from
http://www.gamesparks.com/blog/game-engine-analysis-and-comparison/

Osborn, George: A history of Finland's mobile games industry: It started with Snake (2015), retrieved on 27.05.2016, from
http://www.pocketgamer.biz/feature/61825/finnish-mobile-game-industry-part-1/

Phonearena: History of mobile gaming (2011), retrieved on 17.05.2016, from
http://www.phonearena.com/news/History-of-mobile-gaming_id17949

Quill18: Quill18's Unity 3d & Blender Game Programming Tutorials (n.d.), retrieved on 10.05.2016, from
http://quill18.com/unity_tutorials/

Sager, Ira: Before IPhone and Android Came Simon, the First Smartphone (2012), retrieved on 17.05.2016, from
http://www.bloomberg.com/news/articles/2012-06-29/before-iphone-and-android-came-simon-the-first-smartphone

Scirra: Create Games with Construct 2 (n.d.), retrieved on 25.05.2016, from
https://www.scirra.com/construct2

Scirra: Instant Preview (n.d.), retrieved on 25.05.2016, from
https://www.scirra.com/construct2

Scirra Store: Make Games with Construct 2 (n.d.), retrieved on 29.05.2016, from
https://www.scirra.com/store/construct-2

Sweeney, Tim: IF YOU LOVE SOMETHING, SET IT (2015), retrieved on 25.05.2016, from
https://www.unrealengine.com/blog/ue4-is-free

Unger, Kimberly & Novak, Jeannie: Game Development Essentials: Mobile Game Development (2011)
Yhdysvallat, Kentucky, Independence, ISBN: 9781418052652

Unity Asset Store (n.d.), retrieved on 25.05.2016, from
https://www.assetstore.unity3d.com/en/

Unity Community (n.d.), retrieved on 25.05.2016, from
https://unity3d.com/community

Unity EULA: UNITY PRO AND UNITY PERSONAL SOFTWARE LICENSE AGREEMENT 5.X. (n.d.), retrieved on 17.05.2016, from
https://unity3d.com/legal/eula

Unity: Get Unity (n.d.), retrieved on 27.05.2016, from https://unity3d.com/get-unity

Unity Learn: Learn With Unity (n.d.), retrieved on 25.05.2016, from https://unity3d.com/learn

Unity Manual: Android SDK Setup. (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/android-sdksetup.html

Unity Manual: Canvas Scaler (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/script-CanvasScaler.html

Unity Manual: Creating and Using Scripts (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/CreatingAndUsingScripts.html

Unity Manual: Downloading and Installing Unity. (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/InstallingUnity.html

Unity Manual: Prefabs (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/Prefabs.html

Unity Manual: Rigidbody 2D (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/class-Rigidbody2D.html

Unity Manual: Scenes. (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/CreatingScenes.html

Unity Manual: Troubleshooting Android development. (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/Manual/TroubleShootingAndroid.html

Unity Script Reference: GameObject. (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/ScriptReference/GameObject.html

Unity Script Reference: Mathf.Lerp (n.d.), retrieved on 26.05.2016, from https://docs.unity3d.com/ScriptReference/Mathf.Lerp.html

Unity Script Reference: MonoBehaviour (n.d.), retrieved on 27.05.2016, from http://docs.unity3d.com/ScriptReference/MonoBehaviour.html

Unity Script Reference: PlayerPrefs (n.d.), retrieved on 26.05.2016, from http://docs.unity3d.com/ScriptReference/PlayerPrefs.html

Unity Script Reference: Welcome to the Unity Scripting Reference!, (n.d.), retrieved on 25.05.2016, from http://docs.unity3d.com/ScriptReference/

Unity: UNITY DOWNLOAD ARCHIVE (n.d.), retrieved on 26.05.2016, from
https://unity3d.com/get-unity/download/archive

Unreal Engine: Mobile Game Development. (n.d.), retrieved on 25.05.2016, from
https://docs.unrealengine.com/latest/INT/Platforms/Mobile/

Wikipedia iPhone (1st generation) (n.d.), retrieved on 20.05.2016, from https://en.wikipedia.org/wiki/IPhone_(1st_generation)

Wikipedia: Nokia 7110 (n.d.), retrieved on 17.05.2016, from https://en.wikipedia.org/wiki/Nokia_7110

Wikipedia: Unreal (n.d.), retrieved on 20.05.2016, from https://en.wikipedia.org/wiki/Unreal

Wikipedia: Unreal Engine – Unreal Development Kit. (n.d.), retrieved on 25.05.2016, from
https://en.wikipedia.org/wiki/Unreal_Engine#Unreal_Development_Kit

Wright, Chris: A Brief History of Mobile Games: 2007/8 - Thank God for Steve Jobs (2009), retrieved on 17.05.2016, from
http://www.pocketgamer.biz/feature/10723/a-brief-history-of-mobile-games-20078-thank-god-for-steve-jobs/